WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences

Search Results -

Term	Documents
OFF-CHIP.DWPI,TDBD,EPAB,JPAB.	1003
OFFCHIP.DWPI,TDBD,EPAB,JPAB.	4
MICROCODE.DWPI,TDBD,EPAB,JPAB.	2432
MICRO-CODE.DWPI,TDBD,EPAB,JPAB.	256
((OFFCHIP OR OFF-CHIP) NEAR (MICRO-CODE OR MICROCODE)).JPAB,EPAB,DWPI,TDBD.	2

Database:

- US Patents Full-Text Database
- US Pre-Grant Publication Full-Text Database
- JPO Abstracts Database
- EPO Abstracts Database
- Derwent World Patents Index
- IBM Technical Disclosure Bulletins

Refine Search:

```
(off-chip or offchip) near (microcode or
micro-code)
```

Clear

Search History

Today's Date: 9/24/2001

DB Name	Query	Hit Count	Set Name
JPAB,EPAB,DWPI,TDBD	(off-chip or offchip) near (microcode or micro-code)	2	L21
JPAB,EPAB,DWPI,TDBD	l18 and l19	3	L20
JPAB,EPAB,DWPI,TDBD	(external) near (microcode or micro-code)	18	L19
JPAB,EPAB,DWPI,TDBD	(internal) near (microcode or micro-code)	17	L18
USPT,PGPB	l16 not l11	8	L17
USPT,PGPB	l14 and l15	12	L16
USPT,PGPB	(internal) near (microcode or micro-code)	98	L15
USPT,PGPB	(external) near (microcode or micro-code)	51	L14
USPT,PGPB	(off-chip or offchip) near (microcode or micro-code)	4	L13
USPT,PGPB	(off-chip or offchip) near microcode	4	L12
USPT,PGPB	l9 and l10	4	L11
USPT,PGPB	internal microcode	79	L10
USPT,PGPB	external microcode	30	L9
DWPI	l6 and l7	0	L8
DWPI	external microcode	2	L7
DWPI	internal microcode	2	L6
DWPI	5274829.uref.	0	L5
DWPI	5900025.pn.	1	L4
DWPI	(demers and lentz).in.	1	L3
EPAB	(demers and lentz).in.	1	L2
USPT,PGPB	5274829.pn.	1	L1

WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences

Search Results -

Term	Documents
(28 AND 14).USPT,PGPB.	0

Database:

- US Patents Full-Text Database
- US Pre-Grant Publication Full-Text Database
- JPO Abstracts Database
- EPO Abstracts Database
- Derwent World Patents Index
- IBM Technical Disclosure Bulletins

Refine Search:

```
114 and 128
```

Clear

Search History

Today's Date: 9/24/2001

DB Name	Query	Hit Count	Set Name
USPT,PGPB	l14 and l28	0	L29
USPT,PGPB	((712/24)!.CCLS.)	112	L28
USPT,PGPB	l14 and l26	1	L27
USPT,PGPB	((712/23)!.CCLS.)	553	L26
USPT,PGPB	l24 not l16	3	L25
USPT,PGPB	l14 and internal rom	5	L24
USPT,PGPB	(((712/246)!.CCLS.))	26	L23
USPT,PGPB	((712/248)!.CCLS.)	136	L22
JPAB,EPAB,DWPI,TDBD	(off-chip or offchip) near (microcode or micro-code)	2	L21
JPAB,EPAB,DWPI,TDBD	l18 and l19	3	L20
JPAB,EPAB,DWPI,TDBD	(external) near (microcode or micro-code)	18	L19
JPAB,EPAB,DWPI,TDBD	(internal) near (microcode or micro-code)	17	L18
USPT,PGPB	l16 not l11	8	L17
USPT,PGPB	l14 and l15	12	L16
USPT,PGPB	(internal) near (microcode or micro-code)	98	L15
USPT,PGPB	(external) near (microcode or micro-code)	51	L14
USPT,PGPB	(off-chip or offchip) near (microcode or micro-code)	4	L13
USPT,PGPB	(off-chip or offchip) near microcode	4	L12
USPT,PGPB	l9 and l10	4	L11
USPT,PGPB	internal microcode	79	L10
USPT,PGPB	external microcode	30	L9
DWPI	l6 and l7	0	L8
DWPI	external microcode	2	L7
DWPI	internal microcode	2	L6
DWPI	5274829.uref.	0	L5
DWPI	5900025.pn.	1	L4
DWPI	(demers and lentz).in.	1	L3
EPAB	(demers and lentz).in.	1	L2
USPT,PGPB	5274829.pn.	1	L1

WEST

□ Generate Collection

L17: Entry 1 of 8 File: USPT May 22, 1990

US-PAT-NO: 4928223
DOCUMENT-IDENTIFIER: US 4928223 A

TITLE: Floating point microprocessor with directable two level microinstructions

DATE-ISSUED: May 22, 1990

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Dao; Tich T.	Cupertino	CA	N/A	N/A
Burke; Gary R.	Cupertino	CA	N/A	N/A

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Fairchild Semiconductor Corporation	Cupertino	CA	N/A	N/A	02

APPL-NO: 6/ 901446
DATE FILED: August 28, 1986

PARENT-CASE:
This application is a continuation-in-part of patent application Ser. No. 433,059, filed October 6, 1982.

INT-CL: [5] G06F 9/00, G06F 9/22, G06F 9/26, G06F 13/00
US-CL-ISSUED: 364/200; 364/232.8, 364/243, 364/243.3, 364/244.6, 364/247, 364/247.6, 364/262.4, 364/262.7, 364/262.8
US-CL-CURRENT: 712/247
FIELD-OF-SEARCH: 364/2MSFile, 364/9MSFile

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
□	3631405	December 1971	Hoff et al.	N/A
□	3748649	July 1973	McEowen et al.	340/172
□	3766532	October 1973	Liebel, Jr.	340/172
□	3868649	February 1975	Sato et al.	N/A
□	3886523	May 1975	Ferguson et al.	340/172
□	4032895	June 1977	Lanza	364/200
□	4138718	February 1979	Toke	364/200
□	4153937	May 1979	Poland	N/A
□	4156279	May 1979	Wilhite	364/200
□	4173041	October 1979	Dvorak et al.	364/200
□	4224676	September 1980	Appelt	364/712
□	4307445	December 1981	Tredennick et al.	364/200
□	4323964	April 1982	Gruner	364/200
□	4325121	April 1982	Gunter et al.	364/200
□	4334284	June 1982	Wong	364/748
□	4338661	July 1982	Tredennick	364/200
□	4338675	July 1982	Palmer et al.	364/748
□	4346436	August 1982	Wise	N/A
□	4399505	August 1983	Druke et al.	364/200
□	4434462	February 1984	Guttag et al.	364/200
□	4455604	June 1984	Ahlstrom et al.	364/200

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
1091355	December 1980	CAX	

OTHER PUBLICATIONS

The Organization of Microprogram Stores, Dasgupta, ACM Computing Surveys, vol. II, No. 1, Mar. 1979, pp. 39-65.
Microprogramming: A Tutorial and Survey of Recent Developments, Rauscher et al., IEEE Trans. on Computers, vol. C-29, No. 1, 1/80, pp. 2-19.
Architecture for VLSI-Circuits in Digital Signal Processing, Block et al., Proceedings of the IEEE International Conference on Circuits and Computers, vol. 2 of 2, Oct. 1-3, 1980, N.Y., pp. 1184-1187.
Speedup Arithmetic Functions While Using Less Software, Gordon, Electrical Design News, vol. 27, No. 11, May, 1982, pp. 167-171.
IBM Technical Disclosure Bulletin, vol. 22, No. 8B, Jan. 1980, entitled "Hibride Control Storage".
Article entitled "A Multiprocessor Architecture Adapted to VLSI Custom Design", J. M. C. Alves Marques, Microprocessor Systems: Software, Firmware and Hardware, Six Euromicro Symposium and Microprocessing and Microprogramming, Sep. 16-18, 1980, London, pp. 321-327, North-Holland Publishing Company, New Yor, Oxford.

ART-UNIT: 232
PRIMARY-EXAMINER: Eng; David Y.
ASSISTANT-EXAMINER: Chan; Emily Y.

ATTY-AGENT-FIRM: Patch; Lee Glenn; Michael Murray; William H.

ABSTRACT:

A microprocessor integrator circuit includes split nanocode memories which enables simultaneous execution of an arithmetic operation and an operand fetch for maximizing through-put. The circuit also includes a shared sequencing arithmetic logic unit which handles all microcode sequencing plus memory address sequencing. The circuit also provides nanocode sequencing which enables storage of constants and data in a microcode space which can include an off-chip writable control store. In addition, two level microcode is utilized to enable long routines to be vertically encoded without the overhead of a large number of read only memory outputs.

7 Claims, 29 Drawing figures

□ Generate Collection

L17: Entry 1 of 8 File: USPT May 22, 1990

DOCUMENT-IDENTIFIER: US 4928223 A
TITLE: Floating point microprocessor with directable two level microinstructions

BSPR:
External microcode has also been implemented in bit slice 4-bit microprocessors where it is easier to implement, since all the microcode is off the chip.

BSPR:
The attractiveness of implementing external microcode in microprocessor integrated circuits is thus appreciated in the prior art. However, despite considerable development to date, there remains a need for further improvement in techniques for implementing such external microcode in order to allow user specification of microcode, without long delay cycles inherent in having user microcode implemented in the microprocessor integrated circuit itself during its fabrication.

DRPR:
FIG. 15 is a block diagram of another portion of the microprocessor integrated circuit shown in FIGS. 1 and 2 depicting how the microprocessor of the present invention is connected to an external microcode read only memory.

DEPR:
For use without external microcode, a microprocessor in accordance with FIG. 1 may be provided in a standard 40 pin integrated circuit package. In addition, an extended pin version of the microprocessor integrated circuit may be provided with extra pins for off chip user microcode. The pins have the significance shown in Table 1 below.

DEPR:
FIG. 15 shows how the microprocessor of this invention is connected to an external microcode ROM 3000, in order to allow user supplied microcode to be handled by the microprocessor in the same manner as microcode in the internal ROM 120. Decode PLA 3002 receives instructions on line 3004, decodes those instructions, and provides signals corresponding to the addresses of those instructions on line 3006 to sequence registers 3008. The addresses are supplied by sequence registers 3008 on line 3010 to microcode ROM 120, which outputs the instructions corresponding to the addresses on line 3012 to multiplexer 3014, which in turn supplies the instructions to micro-instruction register 3016. The output of sequence registers 3008 is also connected by line 3018 to the external microcode ROM 3000, located off chip from the microprocessor. External microcode ROM 3000 is also connected to mutliplexer 3014 by line 3020, to supply externally stored micro-instructions to the micro-instruction register 3016. A first portion 3022 of the micro-instruction register 3016 is connected to sequence register 3008 by line 3024. Second and third portions 3026 and 3028 of the micro-instruction register 3016 are respectively connected to nanocode 2 PLA 132 and nanocode 1 ROM 128 by lines 3030 and 3032. The nanocode 2 PLA 132 and the nanocode 1 ROM 128 are respectively connected by lines 3034 and 3036 to nanoinstruction registers 3038 and 3040. Lines 3042 and 3044 provide feedback loops between the nanoinstruction registers 3038 and 3040, respectively, and the nanocode 2 PLA 132 and the nanocode 1 ROM 128. Sequencing register control output signals are supplied by nanoinstruction register 3038 on line 3046. Data path control signals are supplied by nanoinstruction register 3040 on line 3048. Branch inputs are supplied by nanoinstruction register 3040 on line 3050 to branch PLA 3052. Branch condition signals are also supplied on line 3054 to the PLA. Branch control signals are supplied by the branch PLA 3052 on line 3056, for supply to sequence registers 3008. Instruction status signals are supplied to the

nanocode 2 PLA 132 on line 3058.

DEPR:
Sequencing the microcode ROM 120 through use of sequence registers 3008 reduces the number of inputs and outputs of the microcode ROM. Putting an additional level of sequencing around the nanocode allows the external microcode ROM 3000 to be used for constants as well as microinstructions. The control ALU can also be used by address registers as well as the sequence registers 3008 as a result. Further, some microcode reduction is possible by utilizing sequences of nanocodes. The nanocode sequence mechanism is very simple, since branching is handled at the microcode level.

DEPR:
The inclusion of a stack in the microprocessor saves microinstructions, by allowing a microinstruction to "call" a routine. The routine can be a complete instruction, such as floating point addition. Complex instructions, such as tangent, can be built up out of the sequence of simpler instructions, such as polynomial, which are themselves constructed from simpler instructions. As a result, the external microcode ROM 3000 does not need to be either very wide or very long. A width of 16 bits and a length of 4000 bits will handle most applications. Dividing the nanocode store into PLA 132 and ROM 128 reduces its size requirements. The nanocode 2 PLA 132 controls microcode sequencing and memory operations and the nanocode 1 ROM 128 controls the data path in the microprocessor. The sequencing nanocode is very small. Splitting the nanocode in this manner saves repeating non-active outputs throughout the nanocode store, and allows more combinations of sequencing operations, such as CALL/RETURN, with data path operations. The micro-instruction provides an address for both the PLA 132 and the ROM 128, plus an offset which is used to calculate the next microaddress in the case of a branch.

DEPR:
The external microcode ROM 3000 has its own address and data bus, independent of the main CPU bus. The address range of this local bus is limited by the pin out restrictions of an integrated circuit package containing the microprocessor. Assuming a 64-pin package, a reasonable number of pins available for the external microcode ROM, address and data bus is 28, which may be split to give 16 bits of microcode word and 12 bits of address. In this case, only 4K words of external microcode are available. The number of words of external microcode can be increased by using an external random access memory instead of a ROM. The microcode can be downloaded from main memory to configure the microcode according to the nature of the task currently in execution. A further advantage of a local RAM for external microcode storage is that the on chip register storage can be extended using the RAM to store operands without involving the main memory.

DEPR:
As previously described, the instruction takes the form S, D ,T, F; where S is the source (3-bits), D is the destination (3-bits), T is the type (2-bits), and F is the function (8-bits). All ten bits of the T and F fields are input to the decode PLA. The output of this decode is a 13-bit address. The mapping covers both on-chip and off-chip microcode. The most significant bit is a 1 for off-chip addresses. For the on-chip mapping, starting addresses are on 4-word boundaries distributed throughout the microcode address space. The off-chip addresses are defined on 2-word boundaries at the bottom of each 1K block of external microcode.

DEPR:
The major features of the preferred embodiment of the control scheme of the present invention include the following. Two level microcode which allows long routines to be vertically encoded without the overhead of a large number of ROM outputs. Off-chip writable control store allows large extensions to the basic instruction set. External microcode can be stored in standard high speed ROM/PROM, or can be in RAM and down loaded from a host processor. Split nanocode ROM allows simultaneous execution of an arithmethc operation and an operand fetch to maximize through-put. Shared sequencing ALU handles all microcode sequencing plus memory address sequencing. This simplifies the architecture by enabling the use of a single ALU. Nanocode sequencing allows storage of constants and data in the microcode space, which is important for the off-chip microcode. Stack registers provide a means of utilizing microcode as micro-subroutines inside large functions. Pointer registers provide an efficient way to access data and

constants from main memory or local memory. A microcode status register gives the microcode the ability to alter the register assignments, precision and rounding mode of standard routines, to optimize the use of the microcode. Instruction, control and status registers are loaded or read by the host processor to initialize an operation, to set up parameters of that operation and to check the exception flags.

DEPL:
For use with an external microcode in an off chip ROM or other suitable memory, the microprocessor integrated circuit in accordance with FIG. 1 may be provided on an extended, 64 pin package. In addition to the pins and signals specified in Table I, such a 64 pin package has the additional pins and signals shown below in Table II.

WEST

□ Generate Collection

L17: Entry 2 of 8 File: USPT Jun 24, 1986

US-PAT-NO: 4597041
DOCUMENT-IDENTIFIER: US 4597041 A

TITLE: Method and apparatus for enhancing the operation of a data processing system

DATE-ISSUED: June 24, 1986

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Guyer; James M.	Marlboro	MA	N/A	N/A
Epstein; David I.	Framingham	MA	N/A	N/A
Keating; David L.	Holliston	MA	N/A	N/A
Anderson; Walker	Arlington	MA	N/A	N/A
Veres; James E.	Framingham	MA	N/A	N/A
Kimmens; Harold R.	Hudson	MA	N/A	N/A

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Data General Corp.	Westborough	MA	N/A	N/A	02

APPL-NO: 6/ 441839
DATE FILED: November 15, 1982

INT-CL: [4] G06F 9/24
US-CL-ISSUED: 364/200
US-CL-CURRENT: 712/248; 712/208, 712/245
FIELD-OF-SEARCH: 364/2MSFile, 364/9MSFile

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
□	4157587	June 1979	Joyce et al.	364/200
□	4245302	January 1981	Amdahl	364/200
□	4310879	January 1982	Pandeya	364/200
□	4386399	May 1983	Rasala et al.	364/200
□	4450519	May 1984	Guttag et al.	364/200

ART-UNIT: 237
PRIMARY-EXAMINER: Heckler; Thomas M.
ATTY-AGENT-FIRM: Cechony; Gerald Wall; Joel

ABSTRACT:

A data processing system having separate kernel, vertical and horizontal microcode, separate loading of vertical microcode and a permanently resident kernel microcode, and a soft console with dual levels of capability. The system includes a processor having dual ALC and microcode processors, and an instruction processor. Also included are a processor incorporating a multifunction processor memory, a malfunction nibble shifter, and a high speed look-aside memory control.

8 Claims, 9 Drawing figures

WEST

□ Generate Collection

L17: Entry 2 of 8 File: USPT Jun 24, 1986

DOCUMENT-IDENTIFIER: US 4597041 A
TITLE: Method and apparatus for enhancing the operation of a data processing system

DEPR:
Finally, USCL 500 includes internal microcode control logic USCLC, which USCLC receives and decodes control and instruction inputs from Microinstruction Decode (UID) 522, which will be described below, to control operation of USCL 500.

DEPR:
As previously described, CS 101 implements vertical microcode in a writable control store, that is, VUM 534. A means, described next below, is provided to write vertical microcode from external memory to MEM 102 and from MEM 102 to VUM 534. This means also allows the contents of VUM 534 and and KUM 532 read from VUM 534 or KUM 532 to D Bus 112, for example, to verify microcode residing in VUM 534 or KUM 532 or to be read as a source of literal data. This mechanism operate under microcode control and the functions described may be performed under control of microcode provided from either KUM 532 or VUM 534.

DEPR:
Referring first to CPUP 122, CPUP is a 32 bit processor comprised of 8 four bit Advanced Micro Devices (AMD) 2901C microprocessors connected in parallel. CPUP 122 performs all CS 101 arithmetic operations under microcode control of CU 104. CPUP 122 includes a random access memory (RAM), a shift register/buffer, a register file, an arithmetic and logic unit (ALU), and other registers, shift registers, and multiplexers as needed to perform general purpose data manipulation operations, including arithmetic operations. CPUP 122 further includes internal microcode control, which receives instruction inputs from US 116. CPUP 122 receives two inputs, AREG and BREG from US 116 microcode control output which selects, for certain operations, source and destination registers in CPUP 122's register file. As indicated in FIG. 6, CPUP 122 has a 32 bit data input connected from D Bus 112 and a 32 bit output connected to Y Bus 124. The circuitry comprising CPUP 122 are commercially available components well known to those of ordinary skill in the art, and will not be described further except as required for a more thorough understanding of CS 101 during the following detailed descriptions of other portions of PU 106.

WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences

Search Results -

Term	Documents
(16 NOT 11).USPT,PGPB.	8

Database:

- US Patents Full-Text Database
- US Pre-Grant Publication Full-Text Database
- JPO Abstracts Database
- EPO Abstracts Database
- Derwent World Patents Index
- IBM Technical Disclosure Bulletins

Refine Search:

```
l16 not l11
```

Clear

Search History

Today's Date: 9/24/2001

DB Name	Query	Hit Count	Set Name
USPT,PGPB	l16 not l11	8	L17
USPT,PGPB	l14 and l15	12	L16
USPT,PGPB	(internal) near (microcode or micro-code)	98	L15
USPT,PGPB	(external) near (microcode or micro-code)	51	L14
USPT,PGPB	(off-chip or offchip) near (microcode or micro-code)	4	L13
USPT,PGPB	(off-chip or offchip) near microcode	4	L12
USPT,PGPB	l9 and l10	4	L11
USPT,PGPB	internal microcode	79	L10
USPT,PGPB	external microcode	30	L9
DWPI	l6 and l7	0	L8
DWPI	external microcode	2	L7
DWPI	internal microcode	2	L6
DWPI	5274829.uref.	0	L5
DWPI	5900025.pn.	1	L4
DWPI	(demers and lentz).in.	1	L3
EPAB	(demers and lentz).in.	1	L2
USPT,PGPB	5274829.pn.	1	L1

WEST

End of Result Set

□ Generate Collection

L21: Entry 2 of 2 File: TDBD Aug 1, 1982

TDB-ACC-NO: NA8208952

DISCLOSURE TITLE: On-Chip Microcoding of a Microprocessor with only Most Frequently Used Instructions of a Large System. August 1982.

PUBLICATION-DATA:
IBM Technical Disclosure Bulletin, August 1982, US

VOLUME NUMBER: 25
ISSUE NUMBER: 3A
PAGE NUMBER: 952 - 953

PUBLICATION-DATE: August 1, 1982 (19820801)

CROSS REFERENCE: 0018-8689-25-3A-952

DISCLOSURE TEXT:

2p. The performance of a processor is strongly influenced by the length of its critical path from the control store, to the data flow, to the conditional next microword address generator, and back to the control store. A microprocessor, which contains the entire critical path within one chip, can have a very short critical path, and hence a short microcycle time, without expending large amounts of power driving control store addresses and microwords across chip pins. However, currently foreseen microprocessors are limited to an on-chip control store size that will contain only about one quarter of the microcode necessary to implement a mainframe architecture. - The necessary microcode can be partitioned with only the most commonly used microcode, for fixed point and branching instructions, on chip. In a typical instruction mix, these instructions account for 95 percent by frequency of occurrence, and for 60 percent to 75 percent by execution time. Thus, it is very useful to have these instructions implemented by on-chip microcode with a fast cycle time. However, it is also necessary to provide a data path by which microwords can be brought onto the microprocessor chip, and provide an off-chip control store to the chip to implement instructions and functions whose microcode will not fit on the microprocessor itself. - This could be done in several cycles using the existing address and/or data pins for the microword bits, or it could be done using dedicated pins. The off-chip control store must be wide enough for both the microword bits and microword selection bits that are required by the sequencer. The off-chip microword sequencer must have access to on-chip status information, in order to perform conditional microprogram branches and in order to pass control back end forth between on-chip and off-chip functions and instructions. - This method of partitioning the microcode necessary for implementing a large system or mainframe architecture has the the following advantages: A) An architecture of unlimited complexity can be implemented by a sufficiently large off-chip control store. B) Difficult parts of the architecture can be placed off-chip, where they can be corrected without altering the microprocessor chip itself. - product uses a microprocessor chip with larger on-chip control store. D) With care, patches to the on-chip microcode can be implemented in the off-chip microcode if errors are found. E) Since off-chip instructions are executed in the same engine as on-chip instructions, they have full access to registers, condition code and other facilities of the machine. F) All accesses to main storage and channels are made by the same microprocessor. - The arrangement for partitioning microcode between on-chip and off-chip control stores allows the most frequently used instructions to run with the cost/performance of microprocessors (due to the short critical path produced by on-chip microcode), and runs the rest of the architecture's instructions and functions with the cost/performance characteristic of bit slices (with the longer functions with the cost/performance characteristic of bit slices (with the longer critical path produced by off-chip microcode). This is illustrated in the above drawing.

WEST

□ Generate Collection

L25: Entry 1 of 3 File: USPT Oct 31, 2000

US-PAT-NO: 6141740
DOCUMENT-IDENTIFIER: US 6141740 A

TITLE: Apparatus and method for microcode patching for generating a next address

DATE-ISSUED: October 31, 2000

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Mahalingaiah; Rupaka	Austin	TX	N/A	N/A
Miller; Paul K.	Austin	TX	N/A	N/A

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Advanced Micro Devices, Inc.	Sunnyvale	CA	N/A	N/A	02

APPL-NO: 8/ 808483
DATE FILED: March 3, 1997

PARENT-CASE:
CROSS REFERENCE TO RELATED APPLICATIONS This application is related to the following patent applications: "A Byte Queue Divided into Multiple Subqueues for Optimizing Instruction Selection Logic", U.S. patent application Ser. No. 08/650,940, now U.S. Pat. No. 5,748,978. "A Method For Concurrently Dispatching Microcode And Directly-Decoded Instructions In A Microprocessor", filed on Jul. 24, 1996 by Narayan et al., Ser. No. 08/685,656, now abandoned in favor of FWC application Ser. No. 08/878,228.

INT-CL: [7] G06F 12/10, G06F 12/06
US-CL-ISSUED: 711/215; 711/214, 711/202, 711/219, 711/125, 711/144, 711/145
US-CL-CURRENT: 711/215; 711/125, 711/144, 711/145, 711/202, 711/214, 711/219
FIELD-OF-SEARCH: 711/102, 711/103, 711/115, 711/127, 711/157, 711/116, 711/215, 711/214, 711/202, 711/219, 711/125, 711/144, 711/145, 365/230.01

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
□	4028678	June 1977	Moran	340/172.5
□	4028679	June 1977	Divine	N/A
□	4028683	June 1977	Divine et al.	N/A
□	4028684	June 1977	Divine et al.	340/172.5
□	4044338	August 1977	Wolf	N/A
□	4400798	August 1983	Francis et al.	N/A
□	4453212	June 1984	Gaither et al.	N/A

□	4482953	November 1984	Burke	N/A
□	4503492	March 1985	Pilat	711/202
□	4542453	September 1985	Patrick et al.	714/8
□	4577319	March 1986	Takeuchi et al.	714/765
□	4610000	September 1986	Lee	N/A
□	4751703	June 1988	Picon et al.	714/8
□	4802119	January 1989	Heene et al.	N/A
□	4807115	February 1989	Torng	N/A
□	4858105	August 1989	Kuriyama et al.	N/A
□	4928223	May 1990	Dao et al.	N/A
□	4982360	January 1991	Johnson et al.	N/A
□	5053631	October 1991	Perlman et al.	N/A
□	5053949	October 1991	Allison et al.	N/A
□	5058048	October 1991	Gupta et al.	N/A
□	5129067	July 1992	Johnson	N/A
□	5136697	August 1992	Johnson	N/A
□	5226126	July 1993	McFarland et al.	N/A
□	5226130	July 1993	Favor et al.	N/A
□	5295247	March 1994	Chang et al.	N/A
□	5321830	June 1994	Nakamura et al.	N/A
□	5333292	July 1994	Takemoto et al.	711/103
□	5355463	October 1994	Moeller	711/203
□	5367571	November 1994	Bowen et al.	N/A
□	5379301	January 1995	Sato et al.	N/A
□	5425036	June 1995	Liu et al.	N/A
□	5440632	August 1995	Bacon et al.	N/A
□	5454100	September 1995	Sagane	N/A
□	5481713	January 1996	Wetmore et al.	N/A
□	5694348	December 1997	Guttag et al.	708/525
□	5694587	December 1997	Webb et al.	712/224
□	5713035	January 1998	Farrell et al.	712/41
□	5790843	August 1998	Borkenhagen et al.	N/A
□	5790860	August 1998	Wetmore et al.	N/A
□	5794063	August 1998	Favor	N/A
□	5796972	August 1998	Johnson et al.	N/A
□	5796974	August 1998	Goddard et al.	712/211
□	5799144	August 1998	Mio	N/A
□	5829012	October 1998	Marlan et al.	N/A

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
0259095	March 1988	EPX	
0381471	August 1990	EPX	
0459232	December 1991	EPX	
2263985	August 1993	GBX	
2263987	August 1993	GBX	
2281422	March 1995	GBX	

OTHER PUBLICATIONS

Intel, "Chapter 2: Microprocessor Architecture Overview," pp. 2-1 through 2-4.
Michael Slater, "AMD's K5 Designed to Outrun Pentium," Microprocessor Report, vol. 8, No. 14, Oct. 24, 1994, 7 pages.
Sebastian Rupley and John Clyman, "P6: The Next Step?," PC Magazine, Sep. 12, 1995, 16 pages.
Tom R. Halfhill, "AMD K6 Takes On Intel P6," BYTE, Jan. 1996, 4 pages.
IBM Technical Disclosure Bulletin entitled, "On-Site ROS Patch Mechanism," vol. 30, No. 5, Oct. 1987, New York, U.S.A., pp. 158-160.
Scherpenberg, F.A. et al., Electronics De 1984 A 1985: Electronics Week, "Asynchronous Circuits Accelerate Access to 256-K Read-Only Memory," pp. 141-145.
International Search Report for PCT/US 96/08131.

ART-UNIT: 272
PRIMARY-EXAMINER: Peikari; B. James
ATTY-AGENT-FIRM: Conley, Rose & Tayon, PC Merkel; Lawrence J.

ABSTRACT:

A superscalar microprocessor implements a microcode instruction unit that patches existing microcode instructions with substitute microcode instructions. A flag bit is associated with each line of microcode in the microcode instruction unit. If the flag bit is asserted, the microcode instruction unit branches to a patch microcode routine that causes a substitute microcode instruction stored in external RAM to be loaded into patch data registers. The transfer of the substitute microcode instruction to the patch data registers is accomplished using data transfer procedures. The microcode instruction unit then dispatches the substitute instructions stored in the patch data registers and the substitute instruction is executed by a functional unit in place of the existing microcode instruction.

22 Claims, 7 Drawing figures

WEST

□ Generate Collection

L25: Entry 1 of 3 File: USPT Oct 31, 2000

DOCUMENT-IDENTIFIER: US 6141740 A
TITLE: Apparatus and method for microcode patching for generating a next address

BSPR:
When errors (or bugs) are found in microcode instructions, these errors are documented to system designers. Typically, the system designers run simulations to find ways to change the microcode to correct the errors detected. These changes cannot be effectively tested until the next prototype is produced, with the changes to the microcode embedded in the internal ROM of the subsequent processor prototype. A problem with this approach is that the changes to the microcode cannot be verified in the system environment before the changes are committed to silicon. This procedure can greatly increase the cost and time expended during the design process, as unverified changes are made to the microcode and incorporated in a subsequent prototype of the microprocessor, only to fail.

BSPR:
One way to overcome the above problem is to incorporate a technique for patching existing microcode instructions with substitute microcode instructions. When a microcode instruction that needs to be patched is encountered, the instruction fetching mechanism of the microprocessor accesses the substitute microcode instruction from external memory and loads the substitute instruction into the instruction cache. As used herein, the term "external memory" refers to any storage device external to the microcode unit. The substitute instruction, or patched instruction, is then routed from the instruction cache to the microcode unit. The microcode unit uses the substitute microcode instruction as a substitute for the existing microcode instruction.

WEST

□ Generate Collection

L25: Entry 2 of 3 File: USPT Jul 30, 1991

US-PAT-NO: 5036453
DOCUMENT-IDENTIFIER: US 5036453 A

TITLE: Master/slave sequencing processor

DATE-ISSUED: July 30, 1991

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Renner; Karl	Dallas	TX	N/A	N/A
Shanklin; John P.	Colorado Springs	CO	N/A	N/A

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Texas Instruments Incorporated	Dallas	TX	N/A	N/A	02

APPL-NO: 7/ 390507
DATE FILED: August 2, 1989

PARENT-CASE:
RELATED APPLICATION This is a continuation application of application Ser. No. 06/809,084, filed Dec. 12, 1985 and entitled "MASTER/SLAVE SEQUENCING PROCESSOR", now abandoned.

INT-CL: [5] G06F 15/16
US-CL-ISSUED: 364/200; 364/228.9, 364/230.4, 364/232.21, 364/243.1, 364/262.4, 364/271.2
US-CL-CURRENT: 712/16; 712/31
FIELD-OF-SEARCH: 364/2MSFile, 364/9MSFile

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
□	3537074	October 1970	Stokes et al.	364/200
□	3544973	December 1970	Borck, Jr. et l.	364/200
□	3634830	January 1972	Baskin	364/200
□	3812470	May 1974	Murtha et al.	364/200
□	3962685	June 1976	Belle Isle	364/200
□	4051551	September 1977	Lawrie et al.	364/200
□	4065808	December 1977	Schomberg et al.	364/200
□	4101960	July 1978	Stokes et al.	364/200
□	4128880	December 1978	Cray, Jr.	364/200
□	4302818	November 1981	Niemann	364/736 X
□	4306286	December 1981	Cocke et al.	364/200
□	4344134	August 1982	Barnes	364/200
□	4365292	December 1982	Barnes et al.	364/200
□	4376973	March 1983	Chivers	364/200
□	4412303	October 1983	Barnes et al.	364/900
□	4414624	November 1983	Summer, Jr. et al.	364/200
□	4435758	March 1984	Lorie et al.	364/200
□	4435765	March 1984	Uchida et al.	364/200
□	4523273	June 1985	Adams, III et al.	364/200
□	4541048	September 1986	Propster et al.	364/200
□	4542455	September 1985	Demeure	364/200
□	4574348	March 1986	Scallon	364/200
□	4574394	March 1986	Holsztynski et al.	364/200
□	4589067	May 1986	Porter et al.	364/200
□	4617625	October 1986	Nagashima et al.	364/200
□	4641238	February 1987	Kneib	364/200
□	4745544	May 1988	Renner et al.	364/200
□	4855903	August 1989	Carleton et al.	364/200

OTHER PUBLICATIONS

Intel Co., "8080 Instruction Set," The 8080/8085 Microprocessor Book, pp. 38-54 (1980).

ART-UNIT: 237
PRIMARY-EXAMINER: Shaw; Gareth D.
ASSISTANT-EXAMINER: Fagan; Matthew C.
ATTY-AGENT-FIRM: Grossman; Rene' E. Sharp; Melvin

ABSTRACT:

An array processor includes a master array controller and sequencer (12) and a plurality of slave processors (20a)-(20n). The master generates sequencing commands for sequencing instruction flow in each of the slave processors. The slave processors generate addresses for associated memories (34a)-(34n). The data outputs of the memories are interfaced through a cross point switch (22) to a slave data processor (24). The master (12) is operable to initialize all of the slave devices to a starting address for an internal routine and sequence the instruction flow therein in a synchronous and parallel manner to execute a particular task.

11 Claims, 10 Drawing figures

WEST

□ Generate Collection

L25: Entry 2 of 3 File: USPT Jul 30, 1991

DOCUMENT-IDENTIFIER: US 5036453 A
TITLE: Master/slave sequencing processor

DEPR:
The output of register file 132 is input to a three input multiplexer 140, the output of which is connected to a 16.times.16 data stack 142. The output of data stack 142 is input back to one of the inputs of multiplexer 130 and also to one input of a four input multiplexer 144. The output of multiplexer 144 is connected to address bus 48 and also to the input of a program counter (PC) 146. The multiplexer 144 has sixteen bits output therefrom, eleven of which are input to the PC 146 and all of which are output to address bus 48 for input to the external microcode memory 14. In addition, multiple microcode memories can be utilized with three bits output by the multiplexer 144 being input to a decode circuit 148 for a chip select operation.

DEPR:
The output of the PC 146 is looped back to the input of the multiplexer 144 through an incrementing circuit 150 and also input an internal instruction ROM 152. The output of the incrementing circuit 150 is also input to one input of the multiplexer 140 for input to the data stack 142. The instruction ROM 152 is operable to output both the control word to the control bus 18 and also an op code for internal use by the ACS 12. The op code is input to a two input multiplexer 154, the output of which is input to an instruction register decode circuit 156. The instruction register decode circuit 156 provides all the microcode instructions for controlling the ACS 12. The multiplexer 154 also receives the op code output from the external microcode memory 14 on op code bus 50.

DEPR:
The instruction ROM 152 is an internal instruction ROM as opposed to the external microcode instruction ROM 14. In one mode, multiplexer 154 is controlled to receive only op code information from the external memory 14 with the control bus 18 interfaced with the external memory. In the second mode, the instructions are stored in the internal instruction ROM 152 and output therefrom to the control bus 18.

DEPR:
The instruction ROM 176 has the output thereof connected to a multiplexer 180, the output of which is connected to an instruction register decode circuit 182. In operation, the address output of PC 174 selects a predetermined instruction from ROM 176 for output to decode circuit 182. This provides the microcode instruction for execution. Alternatively, the output of the sequencer can be output to an external microcode ROM (not shown) on a line 184. The instruction code output by the ROM is input to the other input of the multiplexer 180 on a line 186. In this mode of operation, the multiplexer 180 is controlled to select the output from the external ROM instead of from internal ROM 176.

WEST

Help | Logout | Interrupt

Main Menu | Search Form | Posting Counts | Show S Numbers | Edit S Numbers | Preferences

Search Results -

Term	Documents
(24 NOT 16).USPT,PGPB.	3

Database:
US Patents Full-Text Database
US Pre-Grant Publication Full-Text Database
JPO Abstracts Database
EPO Abstracts Database
Derwent World Patents Index
IBM Technical Disclosure Bulletins

Refine Search:

```
l24 not l16
```

Clear

Search History

Today's Date: 9/24/2001

DB Name	Query	Hit Count	Set Name
USPT,PGPB	l24 not l16	3	L25
USPT,PGPB	l14 and internal rom	5	L24
USPT,PGPB	(((712/246)!.CCLS.))	26	L23
USPT,PGPB	((712/248)!.CCLS.)	136	L22
JPAB,EPAB,DWPI,TDBD	(off-chip or offchip) near (microcode or micro-code)	2	L21
JPAB,EPAB,DWPI,TDBD	l18 and l19	3	L20
JPAB,EPAB,DWPI,TDBD	(external) near (microcode or micro-code)	18	L19
JPAB,EPAB,DWPI,TDBD	(internal) near (microcode or micro-code)	17	L18
USPT,PGPB	l16 not l11	8	L17
USPT,PGPB	l14 and l15	12	L16
USPT,PGPB	(internal) near (microcode or micro-code)	98	L15
USPT,PGPB	(external) near (microcode or micro-code)	51	L14
USPT,PGPB	(off-chip or offchip) near (microcode or micro-code)	4	L13
USPT,PGPB	(off-chip or offchip) near microcode	4	L12
USPT,PGPB	l9 and l10	4	L11
USPT,PGPB	internal microcode	79	L10
USPT,PGPB	external microcode	30	L9
DWPI	l6 and l7	0	L8
DWPI	external microcode	2	L7
DWPI	internal microcode	2	L6
DWPI	5274829.uref.	0	L5
DWPI	5900025.pn.	1	L4
DWPI	(demers and lentz).in.	1	L3
EPAB	(demers and lentz).in.	1	L2
USPT,PGPB	5274829.pn.	1	L1

WEST

End of Result Set

□ Generate Collection

L27: Entry 1 of 1 File: USPT Nov 9, 1999

US-PAT-NO: 5983334
DOCUMENT-IDENTIFIER: US 5983334 A

TITLE: Superscalar microprocessor for out-of-order and concurrently executing at least two RISC instructions translating from in-order CISC instructions

DATE-ISSUED: November 9, 1999

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Coon; Brett	San Jose	CA	N/A	N/A
Miyayama; Yoshiyuki	Santa Clara	CA	N/A	N/A
Nguyen; Le Trong	Monte Sereno	CA	N/A	N/A
Wang; Johannes	Redwood City	CA	N/A	N/A

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
Seiko Epson Corporation	Tokyo	N/A	N/A	JPX	03

APPL-NO: 8/ 784339
DATE FILED: January 16, 1997

PARENT-CASE:
CROSS-REFERENCE TO RELATED APPLICATIONS This application is a continuation of application Ser. No. 08/460,272, filed on Jun. 2, 1995, now U.S. Pat. No. 5,619,666, which is a continuation of application Ser. No. 07/857,599, filed Mar. 31, 1992, now U.S. Pat. No. 5,438,668. The following are commonly owned, applications: "A ROM With RAM Cell and Cyclic Redundancy Check Circuit", application Ser. No. 07/802,816, filed Dec. 6, 1991, now abandoned; "High-Performance, Superscalar-Based Computer System with Out-of-Order Instruction Execution", application Ser. No. 07/817,810, filed Jan. 8, 1992 now U.S. Pat. No. 5,539,911. "High-Performance, Superscalar-Based Computer System with Out-of-Order Instruction Execution and Concurrent Results Distribution", Ser. No. 08,397,016 filed Mar. 1, 1995, now U.S. Pat. No. 5,560,032, which is a file wrapper continuation of application Ser. No. 07/817,809, filed Jan. 8, 1992. The disclosures of the above applications are incorporated herein by reference.

INT-CL: [6] G06F 9/30, G06F 9/38
US-CL-ISSUED: 712/23; 712/41, 712/206, 712/215, 712/217
US-CL-CURRENT: 712/23; 712/206, 712/215, 712/217, 712/41
FIELD-OF-SEARCH: 712/23, 712/41, 712/206, 712/215, 712/217

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected | Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
□	3346851	October 1967	Thornton et al.	N/A

□	3771138	November 1973	Celtruda et al.	N/A
□	3789365	January 1974	Jen et al.	N/A
□	3916388	October 1975	Shimp et al.	N/A
□	4034349	July 1977	Monaco et al.	N/A
□	4084235	April 1978	Hirtle	N/A
□	4189768	February 1980	Liptay et al.	N/A
□	4189772	February 1980	Liptay et al.	N/A
□	4200927	April 1980	Highes et al.	N/A
□	4228495	October 1980	Bernhard et al.	N/A
□	4236206	November 1980	Strecker et al.	N/A
□	4296470	October 1981	Fairchild et al.	N/A
□	4315314	February 1982	Russo	N/A
□	4317170	February 1982	Wada et al.	N/A
□	4410939	October 1983	Kawakami	N/A
□	4434461	February 1984	Puhl	N/A
□	4454578	June 1984	Matsumoto et al.	N/A
□	4456955	June 1984	Yanagita et al.	N/A
□	4459657	July 1984	Murao	N/A
□	4476525	October 1984	Ishii	N/A
□	4514803	April 1985	Agnew et al.	N/A
□	4569016	February 1986	Hao et al.	N/A
□	4587612	May 1986	Fisk et al.	N/A
□	4629989	December 1986	Toril	N/A
□	4654781	March 1987	Schwartz et al.	N/A
□	4675806	June 1987	Uchida	N/A
□	4714994	December 1987	Oklobzija et al.	N/A
□	4722049	January 1988	Lahti	N/A
□	4739471	April 1988	Baum et al.	N/A
□	4752873	June 1988	Shonai et al.	N/A
□	4758948	July 1988	May et al.	N/A
□	4766566	August 1988	Chuang	N/A
□	4771376	September 1988	Kamiya	N/A
□	4807115	February 1989	Torng	N/A
□	4814976	March 1989	Hansen et al.	N/A
□	4858105	August 1989	Kuriyama et al.	N/A
□	4897810	January 1990	Nix	N/A
□	4901228	February 1990	Kodama	N/A
□	4903196	February 1990	Pomerene et al.	N/A

□	4924376	May 1990	Ooi	N/A
□	4926323	May 1990	Baror et al.	N/A
□	4942525	July 1990	Shintani et al.	N/A
□	4985825	January 1991	Webb, Jr. et al.	N/A
□	4992931	February 1991	Hirasawa	N/A
□	4992934	February 1991	Portanova et al.	N/A
□	4992938	February 1991	Cocke et al.	N/A
□	5003462	March 1991	Blaner et al.	N/A
□	5101341	March 1992	Circello et al.	N/A
□	5127091	June 1992	Boufarah et al.	N/A
□	5133072	July 1992	Buzbee	N/A
□	5148528	September 1992	Fite et al.	N/A
□	5168571	December 1992	Hoover et al.	N/A
□	5193206	March 1993	Mills	N/A
□	5226126	July 1993	McFarland et al.	N/A
□	5226170	July 1993	Rubinfeld	N/A
□	5230068	July 1993	Van Dyke et al.	N/A
□	5335460	August 1994	Eickemeyer et al.	N/A
□	5390355	February 1995	Horst	N/A
□	5442757	August 1995	McFarland et al.	N/A
□	5487156	January 1996	Popescu et al.	N/A
□	5539911	July 1996	Nguyen et al.	N/A
□	5546552	August 1996	Coon et al.	395/385
□	5561776	October 1996	Popescu et al.	N/A
□	5574927	November 1996	Scantlin	N/A
□	5592636	January 1997	Popescu et al.	N/A
□	5625837	April 1997	Popescu et al.	N/A
□	5627983	May 1997	Popescu et al.	N/A
□	5651125	July 1997	Witt et al.	N/A
□	5689720	November 1997	Nguyen et al.	N/A
□	5708841	January 1998	Popescu et al.	N/A
□	5768575	June 1998	McFarland et al.	N/A
□	5778210	July 1998	Henstrom et al.	N/A
□	5797025	August 1998	Popescu et al.	N/A
□	5832205	November 1998	Kelly et al.	N/A
□	5832293	November 1998	Popescu et al.	N/A

FOREIGN PATENT DOCUMENTS

FOREIGN-PAT-NO	PUBN-DATE	COUNTRY	US-CL
0 136 179 A2	April 1985	EPX	
0 170 398	February 1986	EPX	
0 171 595 A2	February 1986	EPX	
0 272 198	June 1988	EPX	
0 354 585 A2	February 1990	EPX	
0 368 332 A2	May 1990	EPX	
0 372 751 A2	June 1990	EPX	
0 377 991 A3	July 1990	EPX	
0 380 854	August 1990	EPX	
0 402 856 A2	December 1990	EPX	
0 419 105 A2	March 1991	EPX	
0 426 393 A2	May 1991	EPX	
0 461 257	December 1991	EPX	
0 473 420	March 1992	EPX	
0 479 390 A2	April 1992	EPX	
2 575 564 A1	April 1986	FRX	
2 011 682	July 1979	GBX	
2230116	October 1990	GBX	

OTHER PUBLICATIONS

Faix et al., "Combined Macro/Micro Program Machine", IBM Technical Disclosure Bulletin, vol. 14, No. 1, p. 298, Jun. 1971.
Patent Abstracts of Japan, vol. 009, No. 006 (p-326), Jan. 11, 1985.
S. Ohr, "Superlong Instructions Help Supermini Refine Number Crunching", Electronic Design, vol. 33, No. 26, pp. 41-43, Nov. 1985.
McNeley et al., "Emulating a Complex Instruction Set Computer with a Reduced Instruction Set Computer", IEEE Micro, vol. 7, No. 1, pp. 60-71, Feb. 1987.
J. Wharton, "Parallel 486 Pipelines Produce Peak Processor Performance," Microprocessor Report, v3 n6 p. 13(5), Jun. 1989.
S. Garth, "Combining RISC and CISC in PC Systems," IEEE, pp. 10/1-10/5.
i486.TM. Microprocessor Programmer's Reference Manual, Order No. 240486, Intel Corporation, Santa Clara, California, 1990.
J. Hennessy et al., Computer Architecture --A Quantitative Approach, Morgan Kaufman Publishers, Inc. San Mateo, California, 1990.
i486.TM. Microprocessor Hardware Reference ManualOrder No. 240552, Intel Corporation, Santa Clara, California, 1990.
J. Crawford, "The Execution Pipelines of the Intel i486.TM. CPU," IEEE, pp. 254-258, Feb. 26 --Mar. 2, 1990.
M. Johnson, Superscalar Microprocessor Design, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1991.
S. Heath, "Architecture Considerations for a M88000 Superscalar RISC Processor," IEEE, London, England, pp. 1-6, Nov. 4, 1991.
Patent Abstracts of Japan, vol. 017, No. 082 (P-1489), Feb. 18, 1993.
Patent Abstracts of Japan vol. 017, No. 084 (P-1490), Feb. 19, 1993.
"High Performance Dual Architecture Procesor," IBM Technical Disclosure Bulletin, vol. 36, No. 2, Feb. 1993, pp. 231-234.
Bakoglu, H.B. et al.,. "IBM Second-Generation RISC Machine Organization," 1990 IEEE, pp. 173-178.
McGeady, S., "The i960CA SuperScalar Implementation of the 80960 Architecture," 1990 IEEE, pp. 232-240.
Nakano, H., et al., "A 80 Mflops 64-Bit Microprocessor For Parallel Computer," 1991 IEEE pp. 15.2.1--15.2.4.
Andrews, W., "Distinctions Blur Between DSP Solutions," Computer Design, vol. 28, No. 9, May 1, 1989, 11 pages.
Coldwell, R.P. et al., "A VLIW Architecture for a Trace Scheduling Compiler," Proceedings 2.sup.nd Interantional Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS II), Oct. 5-8, 1987, California, pp. 180-192.
Lee, Ruby B., "Precision Architecture," Computer, Jan. 1989, pp. 78-91.
IBM Technical Disclosure Bulletin, "Overlap of Store Multiple Operation With

Succeeding Operations Through Second Set of General Purpose Registers", vol. 28, No. 6, Nov. 1985, New York, NY, pp. 2727-2728.

Jelemensky, J. et al., "New Microcomputer Features Advanced Hardware For Real-Time Control Applications", Electro Conference Record, vol. 14, Apr. 11, 1989, Los Angeles, CA, pp. 1,3,5,7 and 9.

Odnert, D. et al., "Architecture and Compiler Enhancements for PA-RISC Workstations," Proceedings from IEEE Compcon, Feb. 25 --Mar. 1, 1991, San Francisco, CA, pp. 214-218.

Melear, C., Motorola, Inc., "The Design of the 88000 RISC Family," IEEE Micro, vol. 9, No. 2, Apr. 1989, Los Alamitos, CA, pp. 26-38.

Abstract for Publication No. JP60225943, Published Nov. 11, 1985, from Patents Abstracts of Japan, Abstract Vol. No. 010089, p. 1.

Popescu, V. et al., "The Metaflow Architecture," IEEE Micro, Jun. 1991, pp. 10-13 and 63-73.

Smith, J.E. et al., "Implementing Precise Interrupts in Pipelined Processors," IEEE Transactions On Computers, vol. 37, No. 5, May 1988, New York, NY, pp. 562-573.

Weiss, R., "Third-generation of R.I.S. C. processors", EDN, Mar. 30, 1992, pp. 96-108.

8-Bit Embedded Controller Handbook, Intel, 1989, pp. 1-1 --1-19 and 3-1 --3-37.

Smith et al., "Implementation of Precise Interrupts in Pipelined Processors," Proceedings of the 12th Annual International Symposium on Computer Architecture, Jun. 1985, pp. 36-44.

Wedig, R.G., Detection of Concurrency In Directly Executed Language Instructions Streams, (Dissertation), Jun. 1982, pp. 1-179.

Agerwala et al., "High-Performance Reduced Instruction Set Processors," IBM Research Division, Mar. 31, 1987, pp. 1-61.

Gross et al., "Optimizing Delayed Branches,"Proceedings of the 5th Annual Workshop on Microprogramming, Oct. 5-7, 1982, pp. 114-120.

Tjaden et al., "Representation of Concurrency with Ordering Matrices," IEEE Trans. On Computers, vol. C-22, No. 8, Aug. 1973, pp. 752-761.

Tjaden, Representation and Detection of Concurrency Using Ordering Matrices, (Dissertation), 1972, pp. 1-199.

Foster et al., "Percolation of Code to Enhance Parallel Dispatching and Execution," IEEE Trans. On Computers, Dec. 1971, pp. 1412-1415.

Thornton, J.E., Design of a Computer: The Control Data 6600, Control Data Corporation, 1970, pp. 58-140.

Weiss et al., "Instruction Issue Logic in Pipelined Supercomputers," Reprinted from IEEE Trans. on Computers, vol. C-33, No. 11, Nov. 1984, pp. 1013-1022.

Tomasulo, R.M., "An Efficient Algorium for Exploiting Multiple Arithmetic Units," IBM Journal, vol. 11, Jan. 1967, pp. 25-33.

Tjaden, et al. "Detection and Parallel Execution of Independent Instructions," IEEE Trans. On Computers, vol. C-19, No. 10, Oct. 1970, pp. 889-895.

Smith et al., "Limits on Multiple Instruction Issue," Proceedings of the 3rd International Conference on Architectural Support for Programming Languages and Operating Systems, Apr. 1989, pp. 290-302.

Pleszkun et al., "The Performance Potential of Multiple Functional Unit Processors," Proceedings of the 15th Annual Symposium on Computer Architecture, Jun. 1988, pp. 37-44.

Pleszkun et al., "WISQ: A Restartable Architecture Using Queues," Proceedings of the 14th International Symposium on Computer Architecture, Jun. 1987, pp. 290-299.

Patt et al., "Critical Issues Regarding HPS, A High Performance Microarchitecture," Proceedings of the 18th Annual Workshop on Microprogramming, Dec. 1985, pp. 109-116.

Hwu et al., "Checkpoint Repair for High-Performance Out-of-Order Execution Machines," IEEE Trans. On Computers, vol. C-36, No. 12, Dec. 1987, pp. 1496-1514.

Patt et al., "HPS, A New Micoarchitecture: Rationale and Introduction," Proceedings of the 18th Annual Workshop on Microprogramming, Dec. 1985, pp. 103-108.

Keller, R.M., "Look-Ahead Processors,"Computing Surveys, vol. 7, No. 4, Dec. 1975, pp. 177-195.

Jouppi et al., "Available Instruction-Level Parallism for Superscalar and Superpipelined Machines," Proceedings of the 3rd International Conference on Architectual Support for Programming Languages and Operating Sytems, Apr. 1989, pp. 272-282.

Hwu et al., "HPSm, a High Performance Restricted Data Flow Architecture Having Minimal Functionality," Proceedings from ISCA-13, Tokyo, Japan, Jun. 2-5, 1986, pp. 297-306.

Hwu et al., "Exploiting Parallel Microprocessor Microarchitectures with a Compiler Code Generator," Proceedings of the 15th Annual Symposium on Computer Architecture, Jun. 1988, pp. 45-53.
Coldwell, et al., "A VLIW Architecture for a Trace Scheduling Compiler," Proceedings of the 2nd International Conference on Architectural Support for Programming Languages and Operating Systems, Oct. 1987, pp. 180-192.
Uht, A.K., "An Efficient Hardware Algorithm to Extract Concurrency From General-Purpose Code," Proceedings of the 19th Annual Hawaii International Conference on System Sciences, 1986, pp. 41-50.
Charlesworth, A.E., "An Approach to Scientific Array Procesing: The Architectual Design of the AP-120B/FPS-164 Family," Computer, vol. 14, Sep. 1981, pp. 18-27.
Acosta, Ramon D. et al., "An Instruction Issuing Approach to Enhancing Performance in Multiple Functional Unit Processors," IEEE Transactions On Computers, vol. C-35, No. 9, Sep. 1986, pp. 815-825.
Johnson, William M., Super-Scalar Processor Design, (Dissertation), Copyright 1989, 134 pages.
Sohi, Gurindar S. and Sriram Vajapeyam, "Instruction Issue Logic For High-Performance, Interruptable Pipelined Processors," Conference Proceedings of the 14th Annual International Symposium on Computer Architecture, Jun. 2-5, 1987, pp. 27-34.

ART-UNIT: 278
PRIMARY-EXAMINER: Lim; Krisna
ATTY-AGENT-FIRM: Sterne, Kessler, Goldstein & Fox P.L.L.C.

ABSTRACT:

A system and method for extracting complex, variable length computer instructions from a stream of complex instructions each subdivided into a variable number of instructions bytes, and aligning instruction bytes of individual ones of the complex instructions. The system receives a portion of the stream of complex instructions and extracts a first set of instruction bytes starting with the first instruction bytes, using an extract shifter. The set of instruction bytes are then passed to an align latch where they are aligned and output to a next instruction detector. The next instruction detector determines the end of the first instruction based on said set of instruction bytes. An extract shifter is used to extract and provide the next set of instruction bytes to an align shifter which aligns and outputs the next instruction. The process is then repeated for the remaining instruction bytes in the stream of complex instructions. The isolated complex instructions are decoded into nano-instructions which are processed by a RISC processor core.

5 Claims, 24 Drawing figures

WEST

End of Result Set

□ Generate Collection

L27: Entry 1 of 1 File: USPT Nov 9, 1999

DOCUMENT-IDENTIFIER: US 5983334 A
TITLE: Superscalar microprocessor for out-of-order and concurrently executing at least two RISC instructions translating from in-order CISC instructions

DEPR:
On chip microcode routine addressing is handled by the IDU itself The IDU generates 16 bit addresses for accesses to the microcode ROM. If the ROM-invalid bit corresponding to the ROM entry being addressed indicates that the microcode is invalid, the address of external microcode residing off-chip in main memory is calculated. A U.sub.-- Base register holds the upper 16 address bits (called the starting address) of the external microcode residing in main memory. The 16 bit address decoded by the IDU is concatenated with the upper 16 bits in the U.sub.-- Base register to access the external microcode residing in main memory. If the location of the external microcode residing in main memory is changed, the contents of the U.sub.-- Base register can be modified to reflect the new main memory location.

DEPR:
This feature allows microcode updates to be performed by replacing certain routines with alternates in external memory, without forcing all microcode to suffer the reduced performance of external memory accesses. It also makes it possible to remove all ROM from the RISC chip and place the entire microcode in external RAM, to reduce the RISC chip's area requirements or to aid in microcode development

CCOR:
712/23

WEST

Generate Collection

Search Results - Record(s) 1 through 2 of 2 returned.

□ 1. Document ID: US 5991858 A, DE 19508723 A1, WO 9628782 A1, EP 813714 A1, NO 9703728 A, EP 813714 B1, DE 59600813 G, ES 2125100 T3

```
L6: Entry 1 of 2                    File: DWPI                    Nov 23, 1999
```

DERWENT-ACC-NO: 1996-413532
DERWENT-WEEK: 200002
COPYRIGHT 2001 DERWENT INFORMATION LTD

TITLE: Multi-user data processing equipment with memory protection - has memory region access table in operating system memory region with allowed address space for instructions in user memory region

ABTX:
The data processing equipment includes a processor and memory in the form of a portable chip card. The process is adapted with regard to the internal microcode such that the implementation of standard instructions loaded in a user memory region is blocked, which require read or write access to the contents of memory cells).

ABEQ:
The data processing equipment includes a processor and memory in the form of a portable chip card. The process is adapted with regard to the internal microcode such that the implementation of standard instructions loaded in a user memory region is blocked, which require read or write access to the contents of memory cells).

ABEQ:
The data processing equipment includes a processor and memory in the form of a portable chip card. The process is adapted with regard to the internal microcode such that the implementation of standard instructions loaded in a user memory region is blocked, which require read or write access to the contents of memory cells).

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw Desc | Image

□ 2. Document ID: EP 110613 A, AU 8320139 A, BR 8306259 A, CA 1198221 A, CA 1198223 A, US 4569018 A, CA 1204216 A, CA 1204217 A, CA 1204218 A, CA 1204220 A, US 4591972 A, US 4597041 A, CA 1206267 A, IL 70230 A, IL 82688 A, IL 82689 A, EP 110613 B, DE 3382334 G, JP 08016388 A

```
L6: Entry 2 of 2                    File: DWPI                    Jun 13, 1984
```

DERWENT-ACC-NO: 1984-147803
DERWENT-WEEK: 199614

TITLE: High speed, compact digital data processing system - has serial input-output unit providing communication with soft console and another unit for communication with bulk memory devices

ABEQ:
A register is connected to an instruction receiver and a state receiver and is responsive to a sequence control internal microcode control for receiving and storing an initial address of a presently selected one of the microinstruction sequences. A microcode program counter is connected to the register and is responsive to the sequence control internal microcode control for receiving the initial address.

Full | Title | Citation | Front | Review | Classification | Date | Reference | Claims | KWIC | Draw Desc | Image

Generate Collection

Term	Documents
INTERNAL.DWPI.	322257
MICROCODE.DWPI.	441
(INTERNAL ADJ MICROCODE).DWPI.	2

Display 30 Documents, starting with Document: 2

Display Format: KWIC Change Format

WEST

Collections

Definition, Editing, Browsing

Name: Undefined

Contents:

```
DE   19508723A
EP   110613A
```

Comment:

Database:

- US Patents Full-Text Database
- US Patents Pre-Grant Publication Full-Text Database
- JPO Abstracts Database
- EPO Abstracts Database
- Derwent World Patents Index
- IBM Technical Disclosure Bulletins

Save | Save As | Help | Reset | Quit

Print | Search | Get Images | Classification Info

Main Menu | Collection Directory

Logout

WEST

Print Selection

Help Clear Cancel Print

Select?	Patent	Section	Page	Database
☑	DE19508723A	all	all	DWPI
☑	EP110613A	all	all	DWPI

Building	Room	Printer
cpk2 ▼	2b01 ▼	gbkbptr ▼

Main Menu Logout

library home | list alphabetically | list by SIG | search library | register DL | subscribe DL | feedback

acm ACM Digital Library

search results

Page: **1 of 1**
Articles: 1 Ordered By Score

Search: New | Undo | Refine
Order By: Publication | Score | Publication Date
View: Brief Listing | Full Listing | Search Expression | All Articles | +Page Size | -Page Size | Help

No.	Article	Score
1)	Leo: a system for cost effective 3D shaded graphics; Michael F. Deering and Scott R. Nelson; *Proceedings of the 20th annual conference on Computer graphics*, 1993, Pages 101 - 108 [Find Related Articles]	8

go to page: **1 of 1**

library home | list alphabetically | list by SIG | search library | register DL | subscribe DL | feedback

WEST

Generate Collection

Search Results - Record(s) 1 through 4 of 4 returned.

☑ 1. Document ID: US 5500948 A

L11: Entry 1 of 4 File: USPT Mar 19, 1996

DOCUMENT-IDENTIFIER: US 5500948 A
TITLE: Translating instruction pointer virtual addresses to physical addresses for accessing an instruction cache

DEPR:
Refer to FIG. 2. In order to carry out its functions, the IFU contains a ROM (50) which contains the internal microcode used on the chip, an instruction cache (56) to hold instructions fetched from memory, instruction queues (52) for temporary instruction storage between memory and the cache, instruction pre-decode (54) and post-decode logic (58), address translation logic (62), cache tag logic (60), and the necessary IFU control logic (70) to make everything work together.

DEPR:
In order to carry out its functions, the IFU contains a read only memory (ROM-50) which stores an internal microcode used by an execution unit, an instruction cache (56) to hold instructions fetched from memory, instruction queues (52) for temporary instruction storage between memory and the cache, instruction pre-decode (54) and post-decode logic (58), address translation logic (62), cache tags logic (60), and the necessary control logic (70) to make everything work together.

DEPR:
ROM: The ROM (50) contains the internal microcode that are supplied to an execution unit that is not shown in FIG. 1.

DEPR:
The other entry is used to translate "microcode" IPs. Some of the microcode that the execution unit needs is stored in off-chip memory. It is desirable that this external microcode be placed anywhere that the system builder would like. This is allowed by translating the logical microcode IP to the physical address where the external microcode is placed. The single microcode entry in the TWB is used to accomplish this purpose. This microcode entry is only changed under explicit software control. It can never have a "miss". It is really a base address for the location of the external microcode block.

Full | Title | Citation | Front | Review | Classification | Date | Reference   KWIC | Draw Desc | Image

☑ 2. Document ID: US 5423014 A

L11: Entry 2 of 4 File: USPT Jun 6, 1995

DOCUMENT-IDENTIFIER: US 5423014 A
TITLE: Instruction fetch unit with early instruction fetch mechanism

DEPR:
In order to carry out its functions, the IFU contains a ROM (50) which contains

the internal microcode used on the chip, an instruction cache (56) to hold instructions fetched from memory, instruction queues (52) for temporary instruction storage between memory and the cache, instruction pre-decode (54) and post-decode logic (58), address translation logic (62), cache tag logic (60), and the necessary control logic (70) to make everything work together.

DEPR:
ROM: The ROM (50) contains the internal microcode that are supplied to an execution unit that is not shown in FIG. 1.

DEPR:
Bits 16 to 31: These bits are the external microcode base mapping register. When the machine is in external microcode mode they are ORed with bits 0 to 21 (right justified) of the logical address to generate the physical address for the Tags block and for the PhysAddrq21 bus.

DEPR:
Most of the mechanism is implemented in the TWB. The lower four bits of the IFU control register select the caching state for the four linear regions. When in macrocode mode, the two upper address bits control a MUX which selects the appropriate bit to drive onto the ICacheq01 ICacheq01 signal. When in microcode mode, ICacheq01 is always driven to 1, regardless of the control register bits, because external microcode is guaranteed to not be self-modifying, so it doesn't need the bus snooping functions of the data cache. Signal ICacheq01 is delayed by the appropriate number of phases and sent out on ICacheql1 ICacheql1 to the BCL. An exception occurs if in external microcode mode and oscillating between regions. The value from two cycles ago is used. When doing an early fetch, the ICacheql1 bit is sent out one cycle before the correct microaddress is known, so it is a guess. If the guess is wrong, the early fetch is cancelled, to be retried later, hopefully with a correct guess.

DEPR:
The ROM (50) contains 12 KB of internal microcode used by the chip. It is complementary in nature in order to drive onto the cache bit lines. Like the cache, it is capable of reading out 12 to 16 bytes of data per cycle. Because the ROM is not a power of two in size, and because of pitch matching constraints, the lower 8 KB of the ROM maps onto both sides A and B of the cache, alternating "sides" with increasing addresses, while the upper 4 KB of the ROM only maps onto side A.

DEPR:
The logic updates the LRU bit on replaces as well as on matches to avoid a deadlock perpetual restart case when switching between external microcode and macrocode.

DEPR:
When in microcode mode, if logical address bit 22 is not asserted, the address is considered "internal", and the ROM is accessed. If address bit 22 is asserted, the address is considered "external", and the TWB and Cache are accessed. When executing external microcode, logical address bits 0 through 21, right justified, are combined with a 16-bit left justified quantity stored in the IFU control register. Since bits 16 through 21 of these quantities overlap, they are logically ORed together.

DEPR:
With this arrangement, microcode (or the user or OS) has control over the size and alignment of the external microcode space. It can define a 64 KB external microcode space which need only be aligned on a 64 KB boundary, by using bits 16 through 21 for the base address from the IFU control register. Or, it can define an 4 MB external microcode space which must be aligned on an 4 MB boundary, by using bits 16 through 21 as addressing bits. Any size (and alignment) which is a power or 2 between 64 KB and 4 MB is supported, simply by allocating bits 16 through 21, through usage, to either base address in the IFU control register, or to addressing bits.

Full | Title | Citation | Front | Review | Classification | Date | Reference | KWIC | Drawu Desc | Image

□ 3. Document ID: US 5187799 A

L11: Entry 3 of 4 File: USPT Feb 16, 1993

DOCUMENT-IDENTIFIER: US 5187799 A
TITLE: Arithmetic-stack processor which precalculates external stack address before needed by CPU for building high level language executing computers

DEPR:
When a HLL is compiled to run on a microprocessor, there is normally an intermediate language (IL) generated between the source code generated by the programmer (e.g. "C") and the machine language (e.g. 68020 machine code) which is actually loaded and run by the computer to perform the steps of the program. As depicted in FIG. 4, the designers of high level languages typically design the language and its manner of operation around a virtual computer 26; that is, an imaginary computer set up to perform in the desired manner. The IL is most efficient when it is based on a Reverse Polish machine; that is, on a virtual stack machine wherein, as depicted in FIG. 4, the ALU 12 operates on stacks 28 which are loaded out of the main memory 14. In such case, the operators occur on the stacks in the same order in which they are employed. As a result, they are quicker and easier to run. The basic building blocks of a "typical" virtual stack machine are: an ALU, an arithmetic stack, a return stack, an Instruction Fetch Unit (IFU), a microcode-sequencer and some data and instruction memory. Unfortunately, the virtual stack machine around which the HLL is designed never actually exists in its entirety in the prior art. Instead, it is typically implemented as in the computer 10' of FIG. 5; that is, the stacks 28 only exist as part of the memory 14 and the normal overhead of moving operators, data, etc. from memory into the stacks and from there into the ALU 12 for execution is still present. The Novix NC4000, which was mentioned above, is conceptually the closest to the ASP of the present invention; but, there are some important and dramatic differences. The ASP has no internal microcode or decoding, i.e. microcode directly controls the circuit elements as in a bit sliced computer. The ASP pipeline delay between the CPU and the SPU minimizes external RAM speed requirements. The ASP instructions are completely general; and, the internal data paths are from any element to any other element. The NOVIX, on the other hand, has a fixed set of instructions, most of which are FORTH words, and cannot be added to. The ASP has two stack pointers, which can both have a large variety of offsets, including the top of stack (TOS) (very useful for a fast PICK instruction). The use of two ASP's per system allows more powerful sequencing operation--including the ability to do subroutine calls with zero overhead. Finally, the ASP has 32 bits vs. the NOVIX's 16 bits for data (stack and system); and, 24 bits for stack address vs. 8 bits for the NOVIX.

DEPR:
As discussed briefly earlier herein, FIG. 6 shows a simplified functional block diagram of the ASP 30, with its two major sub-blocks. The CPU 32 contains an ALU 12', a parallel multiplier and the top two stack registers. The SPU 34 has two stack pointers and two adders; allowing the external "stack RAM" to simulate two stacks. The provision of two stack pointers is an important point of novelty in and of itself. The M-latch 44 delays the external microcode (M) by one clock cycle before it reaches the CPU 32. This pipeline allows the stack address to be recalculated and waiting when the CPU 32 operation begins.

Full | Title | Citation | Front | Review | Classification | Date | Reference   KWIC | Draw Desc | Image

☑ 4. Document ID: US 4399505 A

L11: Entry 4 of 4 File: USPT Aug 16, 1983

DOCUMENT-IDENTIFIER: US 4399505 A
TITLE: External microcode operation in a multi-level microprocessor

ABPL:
A data processing system the central processing unit (CPU) of which is responsive to and executes microinstructions generated by the decoding of macroinstructions so as to provide one or more data processing operations. The system is arranged so that such microinstructions can be supplied to the CPU from a CPU-resident microcode decoding logic or from one or more external microcode decoding units. Each of the external units can identify a macroinstruction which it is capable of decoding and includes logic for externally providing one or more microinstructions which result from the decoding process. If an external microcode unit and the CPU-resident decode logic are both capable of such decoding operation, the external unit overrides the CPU decoding logic and controls the decoding operation externally. The external microcode unit includes logic for monitoring the number of microinstructions supplied to the CPU which have not yet been executed by the CPU.

BSPR:
While such system has provision for supplying external microcode information utilizing appropriate software, for example, the most effective way for providing such external microcode information is described with reference to the invention disclosed herein. Accordingly, in order to achieve effective use of both the CPU-resident microcode information and the external microcode information, the invention provides for suitable interface logic which permits the most effective control of the transfer of external microcode information from an external microcode unit. While the invention is applicable to such multi-level microcode systems, its use is not limited thereto and the principles thereof are also applicable to single-level systems.

BSPR:
In accordance with the invention, a unique microcode control interface unit is utilized wherein every macroinstruction supplied, for example, by software to the system is simultaneously decoded by both the CPU-resident microcode control unit and the microcode interface units. A subset of the operational code (OP-CODE) in the macroinstruction identifies which type of microcode control unit (i.e., an external microcode unit or an internal, i.e., a CPU-resident, unit) is required to perform the decoding operation in order to produce the required microinstruction or sequence of microinstructions. If the OP-CODE identifies a situation in which both an external and an internal microcode control unit are capable of performing the decoding operation, appropriate logic is provided by the selected external unit so that the external microcode controller can override the CPU-resident controller and, therefore, can provide the required decoding operation.

DEPR:
In systems which utilize multi-level (e.g., two-level) microcode architecture (as in the above referred to application) and which provide for the supplying of external microcode information, as in the system of FIG. 1, the latter information must be supplied through one or more suitable interface units such as shown by units 13 of the figure. In conventional systems capable of utilizing external microcode operation, the CPU normally requires appropriate logic which responds to software for identifying the macroinstruction to be decoded as one which requires decoding either by the internal microcode architecture or by the external microcode architecture. Once the CPU has identified the macroinstruction as requiring an external microcode decoding procedure, it supplies the macroinstruction to the designated external microcontroller unit which then proceeds to decode the macroinstruction so as to provide the initial microinstruction via the microcode bus.

DEPR:
In many such conventional systems once the initial microinstruction has been provided to the CPU from an external microcode unit, the CPU then utilizes its own internal sequencing logic and microcode control store to provide the subsequent microinstructions of the particular sequence required by the decoded macroinstruction. Control of the macroinstruction decoding process, as well as the process for determining the address of each subsequent microinstruction required in the sequence, effectively, therefore, resides in the CPU.

DEPR:

Should both a specified external microcontroller unit 13 and the CPU-resident decoding units of the processor recognize a macroinstruction bit pattern as being one which it is capable of decoding, an acknowledge (ACK) signal from the microcontroller interface unit is always interpreted by the central processor unit as requiring external microcode decoding, even if the CPU resident processor decoding units could decode such macroinstruction. Accordingly, decoding by the external unit prevails. Such operation is designated as an "external microcode override" condition. If no ACK signal is asserted from any external microcontroller unit, the processor utilizes its own internal macrocode decoding units, as discussed in the aforementioned application of Bernstein et al, for performing the macroinstruction decoding operation.

CLPR:
2. A data processing system in accordance with claim 1 wherein each said external microcode control unit further includes

CLPR:
3. A data processing system in accordance with claims 1 or 2 wherein each said external microcode control unit comprises

CLPR:
4. A data processing system in accordance with claim 3 wherein control means of each said external microcode control unit further includes means responsive to said decoding means and to said central processor unit for indicating the number of microinstructions which have been transmitted to said central processor unit and which have not yet been executed by said central processor unit.

CLPR:
6. A data processing system in accordance with claim 5 wherein said external microcode control unit includes

CLPR:
7. A data processing system in accordance with claim 6 wherein said external microcode control unit further includes

CLPR:
11. A data processing system in accordance with claims 1 or 2 wherein said transmitting means includes bus means interconnecting said external microcode control unit and said central processor unit; and

CLPV:
one or more external microcode control units, each including

CLPV:
and said decoding means further including means responsive to a macroinstruction received from said central processor unit for determining whether said macroinstruction is one which is to be decoded by a selected one of said external microcode control units or one which is to be decoded by said central processor unit.

CLPV:
control means responsive to said decoding means for assuring that said macroinstruction is decoded only by said selected external microcode control unit when said macroinstruction is capable of being decoded either by said selected external microcode control unit or by said central processor unit.

CLPV:
a next address mode field for indicating whether the sequence of microinstructions transmitted to said central processor unit from said external microcode control unit is to be continued or is to be completed upon execution of the current microinstruction.

CLPV:
driving means for placing a microinstruction produced by said external microcode control unit on to said bus means for transmission to said central processor unit;

CLPV:
and further wherein each said external microcode control unit includes

CLPV:
means for activating its own driver means only when said external microcode control unit has been selected to decode said macroinstruction and to provide said sequence of one or more microinstructions.

Full | Title | Citation | Front | Review | Classification | Date | Reference   KWIC | Draw. Desc | Image

Generate Collection

Term	Documents
(10 AND 9).USPT,PGPB.	4

Display 30 Documents, starting with Document: 4

Display Format: KWIC Change Format

WEST

Generate Collection

Search Results - Record(s) 1 through 4 of 4 returned.

□ 1. Document ID: US 5500948 A

L11: Entry 1 of 4 File: USPT Mar 19, 1996

US-PAT-NO: 5500948
DOCUMENT-IDENTIFIER: US 5500948 A

TITLE: Translating instruction pointer virtual addresses to physical addresses for accessing an instruction cache

DATE-ISSUED: March 19, 1996

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Hinton; Glenn J.	Portland	OR	N/A	N/A
Riches, Jr.; Robert M.	Hillsboro	OR	N/A	N/A

US-CL-CURRENT: 711/205; 711/140

Full | Title | Citation | Front | Review | Classification | Date | Reference   KWIC | Draw Desc | Image

---

□ 2. Document ID: US 5423014 A

L11: Entry 2 of 4 File: USPT Jun 6, 1995

US-PAT-NO: 5423014
DOCUMENT-IDENTIFIER: US 5423014 A

TITLE: Instruction fetch unit with early instruction fetch mechanism

DATE-ISSUED: June 6, 1995

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Hinton; Glenn J.	Portland	OR	N/A	N/A
Riches, Jr.; Robert M.	Hillsboro	OR	N/A	N/A

US-CL-CURRENT: 711/3; 711/204, 712/207

Full | Title | Citation | Front | Review | Classification | Date | Reference   KWIC | Draw Desc | Image

---

□ 3. Document ID: US 5187799 A

L11: Entry 3 of 4 File: USPT Feb 16, 1993

US-PAT-NO: 5187799
DOCUMENT-IDENTIFIER: US 5187799 A

TITLE: Arithmetic-stack processor which precalculates external stack address before needed by CPU for building high level language executing computers

DATE-ISSUED: February 16, 1993

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
McAuley; Anthony	Morris Plain	NJ	N/A	N/A
Goodman; Rod	Altadena	CA	N/A	N/A

US-CL-CURRENT: 712/36

Full | Title | Citation | Front | Review | Classification | Date | Reference   KWIC | Draw Desc | Image

□ 4. Document ID: US 4399505 A

L11: Entry 4 of 4   File: USPT   Aug 16, 1983

US-PAT-NO: 4399505
DOCUMENT-IDENTIFIER: US 4399505 A

TITLE: External microcode operation in a multi-level microprocessor

DATE-ISSUED: August 16, 1983

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Druke; Michael B.	Chelmsford	MA	N/A	N/A
Feaver; Richard L.	Sunnyvale	CA	N/A	N/A
Kosior; Stefan	Chepachet	RI	N/A	N/A

US-CL-CURRENT: 712/246; 712/248

Full | Title | Citation | Front | Review | Classification | Date | Reference   KWIC | Draw Desc | Image

Generate Collection

Term	Documents
(10 AND 9).USPT,PGPB.	4

Display 30 Documents, starting with Document: 4

Display Format: Change Format

WEST

End of Result Set

□ Generate Collection

L12: Entry 4 of 4 File: USPT Apr 30, 1985

US-PAT-NO: 4514803
DOCUMENT-IDENTIFIER: US 4514803 A

TITLE: Methods for partitioning mainframe instruction sets to implement microprocessor based emulation thereof

DATE-ISSUED: April 30, 1985

INVENTOR-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY
Agnew; Palmer W.	Owego	NY	N/A	N/A
Buonomo; Joseph P.	Endicott	NY	N/A	N/A
Houghtalen; Steven R.	Endicott	NY	N/A	N/A
Kellerman; Anne S.	Endicott	NY	N/A	N/A
Losinger; Raymond E.	Endicott	NY	N/A	N/A
Valashinas; James W.	Endicott	NY	N/A	N/A

ASSIGNEE-INFORMATION:

NAME	CITY	STATE	ZIP CODE	COUNTRY	TYPE CODE
International Business Machines Corporation	Armonk	NY	N/A	N/A	02

APPL-NO: 6/ 371634
DATE FILED: April 26, 1982

INT-CL: [3] G06F 9/22
US-CL-ISSUED: 364/200
US-CL-CURRENT: 703/26
FIELD-OF-SEARCH: 364/2MSFile, 364/9MSFile

PRIOR-ART-DISCLOSED:

U.S. PATENT DOCUMENTS

Search Selected Search ALL

	PAT-NO	ISSUE-DATE	PATENTEE-NAME	US-CL
□	4128876	December 1978	Ames et al.	364/200
□	4354225	October 1982	Frieder et al.	364/200
□	4356546	October 1982	Whiteside et al.	364/200

OTHER PUBLICATIONS

Microprocessor Implementation of Mainframe Processors by Means of Architecture Partitioning-Agnew et al.-IBM Journal R&D, vol. 26, No. 4, 1962.

ART-UNIT: 232
PRIMARY-EXAMINER: Springborn; Harvey E.

ATTY-AGENT-FIRM: Seinberg; Saul A.

ABSTRACT:

Methods of applying LSI and microprocessors to the design of microprocessor-based LSI implementation of mainframe processors are described. A mainframe instruction set is partitioned into two or more subsets, each of which can be implemented by a microprocessor having special on-chip microcode or by a standard off-the-shelf microprocessor running programs written for that purpose. Alternatively, one or more of the subsets can be implemented by a single microprocessor. In addition, a subset of the partitioned instruction set can be implemented by emulating software, by off-chip vertical or horizontal microcode, or by primitives. But, however partitioning is implemented, the end result thereof is to keep the critical flow paths, associated with the most frequently used instruction subset, as short as possible by constraining them to a single chip. The application of this method requires partitioning that makes each identified high performance subset executable on one microprocessor in the current state of technology, a way to quickly pass control back and forth between all of the microprocessors, a suitable way to pass data back and forth between all of the microprocessors, and a technology in which it is economically feasible to have several copies of a complex data flow and control store mechanism.

27 Claims, 8 Drawing figures

WEST

End of Result Set

□ Generate Collection

L12: Entry 4 of 4 File: USPT Apr 30, 1985

DOCUMENT-IDENTIFIER: US 4514803 A
TITLE: Methods for partitioning mainframe instruction sets to implement microprocessor based emulation thereof

BSPR:
This off-chip microcode partitioning method also requires what LSI technology is least adept at providing, namely, large numbers of pins. The data flow chip needs at least a dozen pins to tell the control store what microword to give it next. Even worse, the data flow chip needs from 16 to 100 pins to receive that control word. A processor using this method is often limited to roughly 16-bit control words, and hence a vertical microprogram that can control only one operation at a time, whereas a far higher performance processor could be designed if a 100-bit control word were available. If available, such 100-bit control words would permit a horizontal microprogram that can control several operations in each micro-cycle and thus perform a given function in fewer cycles. It should be noted that the off-chip microcode partitioning method has been particularly successful when applied to bit-slice processors, in which the data flow is not reduced to a single chip, but rather is a collection of chips, each of which implements a particular group of bits thoughout the data flow. Bit-slice processors usually employ bipolar technologies whose densities are limited by the number of gates available, or the ability to cool them, rather than by the number of pins on the chips. The off-chip microcode partitioning method applies to FET implementations only in more unusual cases where many pins are available and the chip density happens to exactly match the number of gates needed to implement the data flow of a desired processor. The Toshiba T88000 16-bit microprocessor happens to meet these conditions. Such an implementation can be best viewed as a bit-slice design in which the implementable slice width has widened to encompass the entire desired dataflow.

BSPR:
For example, Digital Equipment Corporation needed a low-end implementation of their PDP-11 minicomputer architecture. They chose the off-chip microcode partitioning method. The result was the LSI 11 four-chip set manufactured first by Western Digital Corporation and then by Digital Equipment Corporation itself.

DEPR:
This method of partitioning the microcode necessary for implementing a complex mainframe architecture has the advantage of permitting an architecture of unlimited complexity to be implemented by use of a sufficiently large off-chip control store. Further, difficult parts of the architecture can be placed off-chip, where they can be corrected without altering the microprocessor chip itself. In addition, off-chip microcode written for a product so implemented may be placed on chip, with minimal modifications, if a subsequent product uses a microprocessor chip with larger on-chip control store. With care, patches to the on-chip microcode can be implemented in the off-chip microcode if errors are found. Since off-chip instructions are executed in the same engine as on-chip instructions, they have full access to registers, condition code, and other facilities of the machine yielding other advantages. A final advantage accrues from the fact that all accesses to main storage and channels are made by the same microprocessor.

DEPR:
The arrangement for partitioning microcode between on-chip and off-chip control stores allows the most frequently used instructions to benefit from the cost/performance of microprocessors due to the short critical path produced by

on-chip microcode, and runs the remaining instructions and functions with the cost/performance characteristics of bit slices with the longer critical path produced by off-chip microcode.

DEPR:
Some advantages of this method of partitioning the architecture between on-chip microcode and off-chip emulation code are as follows. An instruction in set C can be simulated with relatively few bus cycles. An "instruction" brought in from the private instruction store by one or two bus cycles, initiates a sequence of many microwords which do not require bus cycles. Constant data needed by difficult instructions or by interrupts such as Translate and Test's implied register, or interrupts' many implied storage addresses, can be brought in easily as immediate fields of "instructions" fetched from the private program store. Such constants may be difficult to introduce by way of on-chip microcode. An architecture of unlimited complexity can be emulated by a sufficiently large private program store, if the codes in sets A and B supply functions or sufficient generality. The private program store can be relatively small, because it stores relatively powerful "instructions" each of which is interpreted by many microwords. This is especially true if powerful branch and subroutine call "Instructions" are used to save space.

DEPR:
The transfer of control from on-chip microcode to an off-chip emulation program need not be limited to the time when an I-cycle completes. On-chip microcode should be allowed to call for simulation of the rest of an instruction whenever it detects an unusual condition, so it does not require high performance, that is difficult to handle and would otherwise consume many valuable on-chip microwords. For example, the on-chip microcode for Move Characters should be able to call an off-chip program if it detects operand overlap.